Note: This is a new, untested POGIL. Let Iris know if there are issues!

Name:______________________________ **Partner:** _________________________

Python Activity 15: Functions as Types

In Python, everything has a type, which has some useful implications for functions!

Learning Objectives
Students will be able to:
Content:
- Predict the output of code using functions as objects

Process:
- Write code that treats functions as objects
- Write code that passes functions as parameters

Prior Knowledge
- Python concepts: functions, lists, type(), expressions, assignment

Critical Thinking Questions:

1. Examine the sample interactive python session below.

Interactive Python

```
0 >>> def square(x):
1 ...     return x*x
2 >>> type(square)
3 <class 'function'>
4 >>> some_func = square
5 >>> some_func(5)
6 25
```

a. What *type* of object is `square(..)`?

____

How do you know?

____

b. What *type* of object is `some_func`?

____

How do you know?

____

c. How is the command `type('square')` different from what we see on line 2? How would the returned value differ? ____

d. What might be returned if we entered the command, `some_func(4)`? ____________

Would the return value be different if we called, `square(4)`? ____________

How do you know?

____

FYI: Everything in Python is an object, which means everything has a **type**, including functions!

2. Examine the sample interactive python session below.

Interactive Python

```
0 >>> def square(num):
1 ...     return num*num
2 >>> def cube(num):
3 ...     return num*num*num
```

```
4 >>> funcs = [square, cube]
5 >>> for operation in funcs:
6 ...     print(operation(4))
7 16
8 64
```

a. What type of objects are `square` and `cube`? ______________________________

What type of object is variable `funcs`? ______________ of ______________

What type of object is loop variable `operation`? ___________________________

What parameter was passed to `operation`? ___________________________

b. How does the output on lines 7 and 8 relate to the parameter value passed to `operation`? ____

c. Where are the functions `square` and `cube` being called?

____

Application Questions: Use the Python Interpreter to check your work

1. Write a function, `perform_computations`, that takes as arguments a number and a list of functions, *returning* a list of all the results.

e.g.

```
>>> perform_computations(3, [square, cube])
[9, 27]
```

```
def perform_computation(num, computation_funcs):
```